"Express Mail" mailing label number EV 304939627 US

Date of Deposit: April 01, 2004

Attorney Docket No. 15472US02

MOTION VECTOR ADDRESS COMPUTER ERROR DETECTION

RELATED APPLICATIONS

[0001] This application claims priority to Provisional Application for Patent, Serial No. 60/540,579, entitled "Motion Vector Address Computer Error Detection", filed January 30, 2004, by Pai, and incorporated herein by reference.

FEDERALLY SPONSORED RESEARCH OR DEVELOPMENT

[0002] [Not Applicable]

[MICROFICHE/COPYRIGHT REFERENCE]

[0003] [Not Applicable]

BACKGROUND OF THE INVENTION

[0004] Common video compression algorithms use compression based on temporal redundancies between pictures in the video. For example, MPEG-2 defines pictures that can be predicted from one other picture (a P-picture), two pictures (a B-picture), or not predicted from another picture at all (an I-picture).

[0005] Portions, known as macroblocks, from B and P pictures are predicted from reference pixels in a reference picture. The reference pixels can be spatially displaced from the macroblock that is predicted therefrom. Accordingly, the macroblock is encoded, along with indicator(s) indicating the spatial displacements of the

reference pixels from the position of the macroblock. The indicator(s) is known as a motion vector.

[0006] During decoding, the motion vectors are used to retrieve the reference pixels. The reference pixels are retrieved from a memory storing the reference frame. The memory storing the reference frame is known as a frame buffer. A motion vector address computer determines the appropriate addresses storing the reference pixels for a macroblock, based on motion vectors.

[0007] The motion vectors and other information are provided to the motion vector address computer as parameters. However, the motion vector addresses can be corrupted for a variety of reasons. Where the parameters are corrupted, and where such corruption is undetected, the motion vector address computer proceeds to attempt calculation of the address for the reference pixels.

[0008] Further limitations and disadvantages of conventional and traditional approaches will become apparent to one of ordinary skill in the art through comparison of such systems with the present invention as set forth in the remainder of the present application with reference to the drawings.

BRIEF SUMMARY OF THE INVENTION

[0009] Described herein is a system and method for error detection in a motion vector address computer.

[0010] In one embodiment, there is presented a circuit for determining addresses for reference pixels. The circuit comprises an input and logic. The input receives parameters comprising a picture type indicator for indicating a type of a picture. The logic determines whether the parameters received by the input are valid.

[0011] In another embodiment, there is presented a method for determining addresses for reference pixels. The method comprises receiving parameters, the parameters comprising a picture type indicator for indicating a type of a picture; determining the validity of the parameters; and calculating one or more addresses after determining the validity of the parameters, and if the parameters are valid.

[0012] In another embodiment, there is presented a video decoder for decoding macroblocks. The video decoder comprises a processor, a motion vector address computer, and a video request manager. The processor decodes a set of parameters. The set of parameters comprises a picture type parameter indicating a type of picture and motion vectors indicating reference pixels associated with the macroblock. The motion vector address computer determines the validity of the set of parameters, and calculates addresses associated with motion vectors if the set of parameters are valid. The video request manager fetches reference pixels at the addresses calculated by the motion vector address

computer, if the motion vector address computer determines that the set of parameters are valid.

[0013] These and other features and advantages of the present invention may be appreciated from a review of the following detailed description of the present invention, along with the accompanying figures in which like reference numerals refer to like parts throughout.

BRIEF DESCRIPTION OF SEVERAL VIEWS OF THE DRAWINGS

[0014] **FIGURE 1a** is a block diagram of an exemplary Moving Picture Experts Group (MPEG) encoding process, in accordance with an embodiment of the present invention.

[0015] **FIGURE 1b** is an exemplary sequence of frames in display order, in accordance with an embodiment of the present invention.

[0016] **FIGURE 1c** is an exemplary sequence of frames in decode order, in accordance with an embodiment of the present invention.

[0017] **FIGURE 2** is a block diagram of an exemplary decoder system in accordance with an embodiment of the present invention;

[0018] **FIGURE 3** is a flow diagram describing the operation of the motion vector address computer in accordance with an embodiment of the present invention;

[0019] **FIGURE 4** is a block diagram of an exemplary video decoder in accordance with an embodiment of the present invention; and

[0020] **FIGURE 5** is a block diagram describing a motion vector address computer in acco`rdance with an embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

[0021] **FIGURE 1a** illustrates a block diagram of an exemplary Moving Picture Experts Group (MPEG) encoding process of video data 101, in accordance with an embodiment of the present invention. The video data 101 comprises a series of frames 103. Each frame 103 comprises two-dimensional grids of luminance Y, 105, chrominance red Cr, 107, and chrominance blue C_b, 109, pixels. The two-dimensional grids are divided into 8x8 blocks, where a group of four blocks or a 16x16 block 113 of luminance pixels Y is associated with a block 115 of chrominance red C_r, and a block 117 of chrominance blue C_b pixels. The block 113 of luminance pixels Y, along with its corresponding block 115 of chrominance red pixels C_r, and block 117 of chrominance blue pixels C_b form a data structure known as a macroblock 111. The macroblock 111 also includes additional parameters, including motion vectors, explained hereinafter. Each macroblock 111 represents image data in a 16x16 block area of the image.

[0022] The data in the macroblocks 111 is compressed in accordance with algorithms that take advantage of temporal and spatial redundancies. For example, in a motion picture, neighboring frames 103 usually have many similarities. Motion causes an increase in the differences between frames, the difference being between corresponding pixels of the frames, which necessitate utilizing large values for the transformation from one frame to another. The differences between the frames may be reduced using motion compensation, such that the transformation from frame to frame is minimized. The idea of motion compensation is

based on the fact that when an object moves across a screen, the object may appear in different positions in different frames, but the object itself does not change substantially in appearance, in the sense that the pixels comprising the object have very close values, if not the same, regardless of their position within the frame. Measuring and recording the motion as a vector can reduce the picture differences. The vector can be used during decoding to shift a macroblock 111 of one frame to the appropriate part of another frame, thus creating movement of the object. Hence, instead of encoding the new value for each pixel, a block of pixels can be grouped, and the motion vector, which determines the position of that block of pixels in another frame, is encoded.

[0023] Accordingly, many of the macroblocks 111 are compared to pixels of other frames 103 (reference frames). When an appropriate (most similar, i.e. containing the same object(s)) portion of a reference frame 103 is found, the differences between the portion of the reference frame 103 (reference pixels) and the macroblock 111 are encoded. The difference between the portion of the reference frame 103 and the macroblock 111, the prediction error, is encoded using the discrete cosine transformation, thereby resulting in frequency coefficients. The frequency coefficients are then quantized and Huffman coded.

[0024] The location of the reference pixels in the reference frame 103 is recorded as a motion vector. The motion vector describes the spatial displacement between the macroblock 111 and the reference pixels. The encoded prediction error and the motion vector form part of the data structure encoding the macroblock 111. In the MPEG-2

standard, the macroblocks 111 from one frame 103 (a predicted frame) are limited to prediction from reference pixels of no more than two reference frames 103. It is noted that frames 103 used as a reference frame for a predicted frame 103 can be a predicted frame 103 from another reference frame 103.

[0025] The macroblocks 111 representing a frame are grouped into different slice groups 119. The slice group 119 includes the macroblocks 111, as well as additional parameters describing the slice group. Each of the slice groups 119 forming the frame form the data portion of a picture structure 121. The picture 121 includes the slice groups 119 as well as additional parameters that further define the picture 121.

[0026] The progressive frame parameter indicates whether the picture has been encoded as a progressive frame. If the bit is set, the picture has been encoded as a progressive frame. If the bit is not set, the picture has been encoded as an interlaced frame.

[0027] The prediction type parameter specifies the prediction type. The prediction type can include frame prediction, field prediction, dual prime, and 16x8 MC. The picture type parameter indicates whether the picture is an I-picture, a P-picture, or a B-picture.

[0028] I_0, B_1, B_2, P_3, B_4, B_5, and P_6, **FIGURE 1b**, are exemplary pictures. The arrows illustrate the temporal prediction dependence of each picture. For example, picture B_2 is dependent on reference pictures I_0, and P_3. Pictures coded using temporal redundancy with respect to exclusively earlier pictures of the video sequence are known as

predicted pictures (or P-pictures), for example picture P_3 is coded using reference picture I_0. Pictures coded using temporal redundancy with respect to earlier and/or later pictures of the video sequence are known as bi-directional pictures (or B-pictures), for example, pictures B_1 is coded using pictures I_0 and P_3. Pictures not coded using temporal redundancy are known as I-pictures, for example I_0. In the MPEG-2 standard, I-pictures and P-pictures are also referred to as reference pictures.

[0029] The foregoing data dependency among the pictures requires decoding of certain pictures prior to others. Additionally, the use of later pictures as reference pictures for previous pictures requires that the later picture be decoded prior to the previous picture. As a result, the pictures may be decoded in a different order than the order in which they will be displayed on the screen. Accordingly, the pictures are transmitted in data dependent order, and the decoder reorders the pictures for presentation after decoding. I_0, P_3, B_1, B_2, P_6, B_4, B_5, **Fig. 1c**, represent the pictures in data dependent and decoding order, different from the display order seen in **Fig. 1b**.

[0030] The pictures are then grouped together as a group of pictures (GOP) 123. The GOP 123 also includes additional parameters further describing the GOP. Groups of pictures 123 are then stored, forming what is known as a video elementary stream (VES) 125. The VES 125 is then packetized to form a packetized elementary sequence. Each packet is then associated with a transport header, forming what are known as transport packets.

[0031] The transport packets can be multiplexed with other transport packets carrying other content, such as another video elementary stream 125 or an audio elementary stream. The multiplexed transport packets form what is known as a transport stream. The transport stream is transmitted over a communication medium for decoding and displaying.

[0032] **FIGURE 2** illustrates a block diagram of an exemplary circuit for decoding the compressed video data, in accordance with an embodiment of the present invention. Data is received and stored in a presentation buffer 203 within a Synchronous Dynamic Random Access Memory (SDRAM) 201. The data can be received from either a communication channel or from a local memory, such as, for example, a hard disc or a DVD.

[0033] The data output from the presentation buffer 203 is then passed to a data transport processor 205. The data transport processor 205 demultiplexes the transport stream into packetized elementary stream constituents, and passes the audio transport stream to an audio decoder 215 and the video transport stream to a video transport processor 207 and then to a MPEG video decoder 209. The audio data is then sent to the output blocks, and the video is sent to a display engine 211.

[0034] The display engine 211 scales the video picture, renders the graphics, and constructs the complete display. Once the display is ready to be presented, it is passed to a video output encoder 213 where it is converted to analog video using an internal digital to analog converter (DAC). The digital audio is converted to analog in an audio digital to analog converter (DAC) 217.

[0035] The decoder 209 decodes at least one picture, I_0, B_1, B_2, P_3, B_4, B_5, P_6... during each frame display period. Due to the presence of the B-pictures, B_1, B_2, the decoder 209 decodes the pictures, I_0, B_1, B_2, P_3, B_4, B_5, P_6... in an order that is different from the display order. The decoder 209 decodes each of the reference pictures prior to each picture that is predicted from the reference picture. For example, the decoder 209 decodes I_0, B_1, B_2, P_3, in the order, I_0, P_3, B_1, and B_2. After decoding I_0, the decoder 209 writes I_0 to a frame buffer 220 and decodes P_3. the frame buffer 220 can comprise a variety of memory systems, for example, a DRAM. The macroblocks of P_3 are encoded as prediction errors with respect to reference pixels in I_0. The reference pixels are indicated by motion vectors that are encoded with each macroblock of P_3. Accordingly, the video decoder 209 uses the motion vectors encoded with the macroblocks of P_3 to fetch the reference pixels. Similarly, the video decoder 209 uses motion vectors encoded with the macroblocks of B_1 and B_2 to locate reference pixels in I_0 and P_3.

[0036] To fetch the reference pixels from I_0, the video decoder 209 calculates the frame buffer 220 addresses storing the reference pixels, based on the motion vectors. A circuit known as a motion vector address computer calculates the frame buffer addresses. The video decoder 209 then fetches the pixels at the addresses calculated by the motion vector address computer in the frame buffer 220.

[0037] It is noted that errors can be introduced into the video elementary stream. As a result, the video data, as well as the encoded parameters can be corrupted or erroneous. In the case of motion vectors, errors or

corrupted data can result in decoding inaccurate motion vectors, detection of additional motion vectors that are not valid (false positive), and the failure to detect valid motion vectors (false negative). Attempting to calculate the addresses of the reference pixels with the corrupted data can have a deleterious affect on performance. Accordingly, the motion vector address computer includes error detection logic that detects invalid parameter sets and prevents further processing of erroneous or corrupted data.

[0038] Referring now to **FIGURE 3**, there is illustrated a flow diagram describing the operation of the video decoder 209 in accordance with an embodiment of the present invention. At 270, the video decoder 209 receives and decodes parameters associated with a picture 121. The parameters can include, for example, the picture type parameter, the progressive frame parameter, and the prediction type parameter. At 275, the video decoder 209 receives an encoded macroblock 111 from the picture for decoding. At 280, the video decoder 209 determines, based on the picture type, prediction type, picture structure, and the number of motion vectors received with the macroblock 111, whether the foregoing parameters are valid. If at 280, the foregoing parameters are valid, the video decoder 209 calculates the address(es) (285) associated with any of the motion vectors, and fetches any reference pixels (290) associated with the motion vectors. If at 280, the foregoing parameters are not invalid, the video decoder 209 bypasses 285 and 290. In either case, 275-295 are repeated for each macroblock 111 in the picture 121 until all the macroblocks 111 in the picture are decoded at 295.

When all of the macroblocks 111 in the picture are decoded at 295, 270-295 are repeated for the next picture.

[0039] Referring now to **FIGURE 4**, there is illustrated a block diagram of an exemplary video decoder 209 in accordance with an embodiment of the present invention. The video decoder 209 comprises a compressed data buffer 302, an extractor 304, a processor 306, a motion vector address computer 308, a video request manager 310, a motion compensator 312, a variable length decoder 314, an inverse quantizer 316, and an inverse discrete cosine transformation module 318.

[0040] The video decoder 209 fetches data from the compressed data buffer 302, via an extractor 304 that provides the fetched data to a processor 306. the video decoder 209 decodes at least one picture per frame display period on a macroblock by macroblock basis. At least a portion of the data forming the picture is encoded using variable length code symbols. Accordingly, the variable length coded symbols are provided to a variable length decoder 314. The portions of the data that can be encoded using variable length codes can include the parameters, such as picture type, prediction type, progressive frame, and the motion vectors, as well as the encoded pixel data (frequency coefficients). The parameters are provided to the processor 306, while the frequency coefficients are provided to the inverse quantizer 316. The inverse quantizer 316 inverse quantizes the frequency coefficients and the IDCT module 318 transforms the frequency coefficients to the pixel domain.

[0041] If the macroblock is from an I-picture, the pixel domain data represents the pixels of the macroblock. If the

macroblock is from a P-picture, the pixel domain data represents the prediction error between the macroblock and reference pixels from one other frame. If the macroblock is from a B-picture, the pixel domain data represents the prediction error between the macroblock and reference pixels from two other frames.

[0042] Where the macroblock is from a P or B picture, the video decoder 209 fetches the reference pixels from the reference frame(s) 103. The reference frame(s) is stored in a frame buffer(s) 220. The processor 306 provides the motion vectors encoded with the macroblock 111 to the motion vector address computer 308. The motion vector address computer 308 uses the motion vectors to calculate the address of the reference pixels in the frame buffer 220.

[0043] When the motion vector address computer 308 calculates the addresses associated with the reference pixels, the video request manager 310 fetches the reference pixels at the addresses calculated by the motion vector address computer 308, via memory controller. The reference pixels are then provided to a motion compensator 312. The motion compensator 312 applies the prediction error from the macroblock 111 to the fetched reference pixels, resulting in the decoded macroblock 111. The video request manager 310 then writes the decoded macroblock 111 to the frame buffer 220, using the memory controller.

[0044] The motion vector address computer 308 includes error detection logic that detects invalid parameter sets and prevents further processing of erroneous or corrupted data. The error detection logic examines the parameters that are provided to the motion vector address computer 308

to determine whether the parameters are valid. Based on the number of motion vectors provided, the type of picture (I, P, or B-picture), the prediction type, and the progressive frame parameters, the error detection logic determines whether the received parameters are valid. If the parameters are not valid, the error detection logic prevents further processing by the motion vector address computer 308.

[0045] Referring now to **FIGURE 5**, there is illustrated a block diagram of an exemplary motion vector address computer 308. The motion vector address computer 308 comprises an input 405 for receiving parameters. The parameters include the picture type, the progressive frame and prediction type parameters, and zero or a number of motion vectors.

[0046] A control register 410 stores the picture type parameter, progressive frame parameter, and the prediction type parameter. Motion vector registers 415 store the motion vectors. The motion vector registers 415 are cleared prior to receiving the motion vectors. The control register 410 comprises a plurality of bits 420, each of which are associated with a corresponding one of the motion vector registers 415. The bits are in a particular state, such as set, based on whether the corresponding motion vector register 415 is loaded with a motion vector.

[0047] Accordingly, the control register 410 includes indicators indicating the picture type, whether the picture is progressive, the prediction type, and the number of motion vectors received (the number of bits 420 that are set). Error detection logic 425 examines the control register 410 and determines, based on the picture type

parameter, progressive parameter, and whether the number of motion vectors received is correct. If the number of motion vectors received is correct, based on the picture type, prediction type, and progressive frame parameters, the error detection logic 425 indicates that the parameters are valid. If the number of motion vectors received is not correct, the error detection logic 425 indicates that the parameters are not valid.

[0048] The following table represents the valid sets of parameters:

Picture Type	Prog. Frame	Pred. Type	#MV
I	Any	Any	0
P	Frame	Frame Prediction	1
		Field Prediction	2
		Dual Prime	4
B	Frame	Frame Prediction	2
		Field Prediction	4
P	Field	Field Prediction	1
		Dual Prime	2
		16x8 MC	2
B	Field	Field Prediction	2
		16x8 MC	4

[0049] The error detection logic 425 can comprises a state machine or appropriate combinatorial logic gates that determine whether the control register 410 contents corresponds to one of the combinations listed in the table above. Where the control register 410 contents corresponds to one of the combinations listed in the table above, the error detection logic 425 determines that the received parameters are valid. Where the control register 410 contents does not correspond to one of the combinations listed in the table above, the error detection logic 425 determines that the received parameters are not valid.

[0050] Where the error detection logic 425 determines that the parameters received are valid, the error detection logic 425 signals the foregoing to the processor 306. If the parameters are valid, the processor commands the arithmetic logic unit 430 to calculate addresses. The arithmetic logic unit 430 receives the motion vectors, via the motion vector registers 415. The arithmetic logic unit 430 calculates the addresses storing the reference pixels indicated by the motion vectors. The addresses calculated by the arithmetic logic unit 430 are provided to the video request manager 310. The video request manager 310 then fetches the reference pixel corresponding to the addresses calculated by the arithmetic logic unit 425.

[0051] Where the error detection logic unit 425 determines that the parameters received are invalid, the error detection logic 425 signals the foregoing to the processor 306. The processor 306 can then take appropriate action.[IPai_Bkar3]the T. [IPai_Bkar4] For example, For exFFin the presence of an error, the processor 306 can withhold commanding the arithmetic logic unit 425 to calculate addresses.In such as

case, the MOTC does not provide addresses to the video request manager.

[0052] The embodiments described herein may be implemented as a board level product, as a single chip, application specific integrated circuit (ASIC), or with varying levels of the decoder system integrated with other portions of the system as separate components. The degree of integration of the decoder system will primarily be determined by the speed and cost considerations. Because of the sophisticated nature of modern processor, it is possible to utilize a commercially available processor, which may be implemented external to an ASIC implementation. Alternatively, if the processor is available as an ASIC core or logic block, then the commercially available processor can be implemented as part of an ASIC device wherein certain functions can be implemented in firmware.

[0053] While the present invention has been described with reference to certain embodiments, it will be understood by those skilled in the art that various changes may be made and equivalents may be substituted without departing from the scope of the present invention. In addition, many modifications may be made to adapt a particular situation or material to the teachings of the present invention without departing from its scope. Therefore, it is intended that the present invention not be limited to the particular embodiment disclosed, but that the present invention will include all embodiments falling within the scope of the appended claims.